

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平10-105372

(43) 公開日 平成10年(1998) 4月24日

(51) Int.Cl.⁸

識別記号

F I

G 0 6 F 5/00

G 0 6 F 5/00

H

H 0 3 M 7/30

H 0 3 M 7/30

Z

審査請求 未請求 請求項の数 5 F D (全 11 頁)

(21) 出願番号 特願平8-281410

(22) 出願日 平成 8 年(1996)10月 2 日

(71) 出願人 591044164

株式会社沖データ

東京都港区芝浦四丁目11番地22号

(72) 発明者 藤岡 恭一

東京都港区芝浦四丁目11番地22号 株式会

社沖データ内

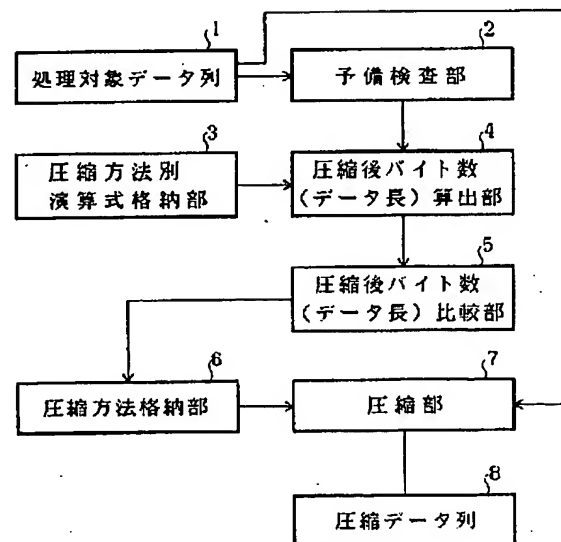
(74) 代理人 弁理士 佐藤 幸男 (外 1 名)

(54) 【発明の名称】 データ列の圧縮装置及び圧縮方法

(57) 【要約】

【解決手段】 予備検査部2は処理対象データ列1を解析して、圧縮後バイト数算出のためのデータを得る。そして、複数の圧縮方式により得られる圧縮データ列のデータ長を算出した結果を圧縮後バイト数比較部5で比較する。こうしてデータ長が最短となる圧縮方法が選択され、改めて処理対象データ列がその方法で圧縮される。

【効果】 実際に全ての方法でデータを圧縮してから、その結果を比較する場合に比べて簡単な演算を用いて予測するから短時間に最適な圧縮方法が決定できる。



本発明の装置の具体例ブロック図

【特許請求の範囲】

【請求項1】 処理対象データ列を、予め用意された複数の圧縮方法で圧縮した場合の、圧縮データ列の長さを計算により予測する圧縮後データ長算出部と、各圧縮方法による圧縮データ列の長さの予測結果を比較して、最適圧縮方法を選択する圧縮後データ長比較部と、選択された圧縮方法で実際に処理対象データ列を圧縮する圧縮部とを備えたことを特徴とするデータ列の圧縮装置。

【請求項2】 処理対象データ列について、予め用意された複数の圧縮方法で圧縮した場合の、圧縮データ列の長さを、実際の圧縮処理をすることなく計算により予測して、その計算結果を比較することにより、最適圧縮方法を選択し、その後、前記処理対象データ列を、選択した最適圧縮方法で圧縮処理することを特徴とするデータ列の圧縮方法。

【請求項3】 請求項2において、RLE圧縮方法による圧縮データ長の予測値は、処理データ列を検査して、ライン長と、繰り返し部分の合計データ長と、繰り返し部分のブロック数とを求めて、繰り返し部分のブロック数と繰り返し部分を表現するための係数長の積と、繰り返さない部分の合計データ長と繰り返さない部分を表現するための係数長の積と、繰り返し部分の合計データ長が一定値以上の場合の係数長の補正分とを加算して求めることを特徴とするデータ列の圧縮方法。

【請求項4】 請求項2において、TIF圧縮方法による圧縮データ長の予測値は、処理データ列を検査して、ライン長と、繰り返し部分の合計データ長と、繰り返し部分のブロック数とを求めて、繰り返し部分のブロック数と繰り返し部分を表現するための係数長の積と、繰り返さない部分の合計データ長と、繰り返さない部分のブロック数と、繰り返し部分の合計データ長が一定値以上の場合の係数長の補正分と、繰り返さない部分の合計データ長が一定値以上の場合の係数長の補正分とを加算して求めることを特徴とするデータ列の圧縮方法。

【請求項5】 請求項2において、Delta-Low圧縮方法による圧縮データ長の予測値は、処理データ列を検査して、ライン長と、前列と異なる部分のブロック数と、前列と異なる部分の合計データ長を求めて、前列と異なる部分のブロック数と、前列と異なる部分の合計データ長と、前列と異なる部分の合計データ長が一定値以上の場合の係数長の補正分とを加算して求めることを特徴とするデータ列の圧縮方法。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、イメージデータのように所定量ずつ入力するデータ列を、その中に含まれる同一データの繰り返しに着目して圧縮する場合に複数種類の圧縮方法を選択しながら最善の圧縮を実行するデータ列圧縮装置及び圧縮方法に関する。

【0002】

【従来の技術】例えば、ホストコンピュータにおいて任意のイメージデータを生成し、これをプリンタへ転送して印刷する場合に、転送すべきデータ量を少なくしてより高速にデータを転送するためにイメージデータの圧縮処理が行われる。また、このような圧縮処理はプリンタ内にホストコンピュータから受信し印刷する直前のデータを一時保存するためのメモリ容量を削減する効果もある。こうしたイメージデータの圧縮を行う場合には、ホストコンピュータに設けられたプリンタドライバがイメージデータを圧縮処理しインタフェースを介してそのイメージデータをプリンタ側に転送する。プリンタはこうして圧縮されたイメージデータを印刷する直前に伸長してから印字ヘッドに供給する。なお、印刷用のイメージデータを圧縮する方法には数種類の方法があり、ラスターライン毎にどの圧縮方法が最適かを調べた上でいずれかの圧縮方法を選択する。

【0003】

【発明が解決しようとする課題】ところで、上記のような従来のデータの圧縮装置や圧縮方法には次のような解決すべき課題があった。複数の圧縮方法が存在する場合に、圧縮処理の対象となるデータの内容によって、どの圧縮方法を採用した場合最もデータ長が短くなるかを調べるために、従来実際に全ての圧縮方法でイメージデータを圧縮し、その結果を比較するようにしていた。

【0004】しかしながら、こういった従来の圧縮方法では、実際に処理対象データを圧縮処理し、何種類かの圧縮方法があれば各圧縮方法により圧縮されたデータをいったん記憶しておくためのメモリが必要となり、最適な圧縮方法を選択するために時間がかかるという問題があった。また、イメージデータを圧縮するために時間がかかれば、ホストコンピュータにおいて印刷命令を発してから実際にプリンタが印刷を開始するまでの時間が長くなるという問題もあった。

【0005】

【課題を解決するための手段】本発明は以上の点を解決するため次の構成を採用する。

〈構成1〉処理対象データ列を、予め用意された複数の圧縮方法で圧縮した場合の、圧縮データ列の長さを計算により予測する圧縮後データ長算出部と、各圧縮方法による圧縮データ列の長さの予測結果を比較して、最適圧縮方法を選択する圧縮後データ長比較部と、選択された圧縮方法で実際に処理対象データ列を圧縮する圧縮部とを備えたことを特徴とするデータ列の圧縮装置。

【0006】〈構成2〉処理対象データ列について、予め用意された複数の圧縮方法で圧縮した場合の、圧縮データ列の長さを、実際の圧縮処理をすることなく計算により予測して、その計算結果を比較することにより、最適圧縮方法を選択し、その後、上記処理対象データ列を、選択した最適圧縮方法で圧縮処理することを特徴とするデータ列の圧縮方法。

【0007】〈構成3〉構成2において、RLE圧縮方法による圧縮データ長の予測値は、処理データ列を検査して、ライン長と、繰り返し部分の合計データ長と、繰り返し部分のブロック数とを求めて、繰り返し部分のブロック数と繰り返し部分を表現するための係数長の積と、繰り返さない部分の合計データ長と繰り返さない部分を表現するための係数長の積と、繰り返し部分の合計データ長が一定値以上の場合の係数長の補正分とを加算して求めることを特徴とするデータ列の圧縮方法。

【0008】〈構成4〉構成2において、TIF圧縮方法による圧縮データ長の予測値は、処理データ列を検査して、ライン長と、繰り返し部分の合計データ長と、繰り返し部分のブロック数とを求めて、繰り返し部分のブロック数と繰り返し部分を表現するための係数長の積と、繰り返さない部分の合計データ長と、繰り返さない部分のブロック数と、繰り返し部分の合計データ長が一定値以上の場合の係数長の補正分と、繰り返さない部分の合計データ長が一定値以上の場合の係数長の補正分とを加算して求めることを特徴とするデータ列の圧縮方法。

【0009】〈構成5〉構成2において、Delta-Low圧縮方法による圧縮データ長の予測値は、処理データ列を検査して、ライン長と、前列と異なる部分のブロック数と、前列と異なる部分の合計データ長を求めて、前列と異なる部分のブロック数と、前列と異なる部分の合計データ長と、前列と異なる部分の合計データ長が一定値以上の場合の係数長の補正分とを加算して求めることを特徴とするデータ列の圧縮方法。

【0010】

【発明の実施の形態】以下、本発明の実施の形態を具体例を用いて説明する。

〈具体例1〉図1は、本発明の装置の具体例を示すブロック図である。この装置は、処理対象データ列1を受け入れて、実際に圧縮処理をする前に所定の検査を行う予備検査部2と、その出力を用いた圧縮後のデータ長予測演算のための式を格納する圧縮方法別演算式格納部3、圧縮後バイト数算出部4、圧縮後バイト数比較部5及び圧縮方法格納部6と、選択された最適の圧縮方法を用いて処理対象データ列1を圧縮する圧縮部7とから構成されている。この圧縮部7の出力する圧縮データ列8が例えばプリンタに送られて印刷される。

【0011】なお、処理対象データ列1は、この例ではプリンタ用のイメージデータとする。もちろん、本発明

はプリンタ用のイメージデータの他、ファクシミリ装置、あるいはイメージデータの送受信やメモリ格納等の際に、圧縮処理することが適する各種のデータに利用できる。

【0012】圧縮方法別演算式格納部3は、メモリから構成され、処理対象データ列1を圧縮するための各種の方法が用意されている場合に、それらの方法を用いてデータを圧縮した場合に、そのデータ長がいかなる値になるかを簡単な計算によって予測するための式が、圧縮方法に対応させて格納してある。予備検査部2は、このような演算に必要なパラメータ等を処理対象データ列1を検査することによって得る部分である。圧縮後バイト数算出部4は、圧縮方法別演算式格納部3を参照しながら圧縮方法毎に所定の演算を行って圧縮後バイト数を計算し予測する部分である。圧縮後バイト数比較部5は、圧縮方法毎の圧縮後バイト数予測値を比較し、圧縮データ列が最短となる圧縮方法を選択する部分である。圧縮方法格納部6は、圧縮後バイト数比較部5によって選択された圧縮方法を圧縮部7に供給する部分である。圧縮部7は、圧縮方法格納部6によって指定された圧縮方法を用いて処理対象データ列1を圧縮して圧縮データ列8を得る部分である。

【0013】以上のように、本発明の装置は、処理対象データ列1を予め用意された圧縮方法で圧縮した場合に、そのデータ長がどのような値になるかを演算によって予測する。この装置の動作を説明するために、具体的な圧縮方法の例を説明する。図2は、RLE圧縮方法の例説明図である。これから説明する例は、処理対象データ列が1バイト単位のデータにより構成され、1バイトずつ処理データ列をその先頭から検査し、同一データが連続している場合に、所定の圧縮を行う。なお、このように1バイトを単位として圧縮を行うようにしているが、一般には1ビットを単位としてもよいし、また1ワードを単位としてもよいし、圧縮のための繰り返し単位は自由に選定できる。

【0014】図2に示す例では、例えば内容が“AA”というデータが4バイト繰り返されている。その次に、“OF”というデータと“FO”というデータが続いている。このデータをRLE圧縮方法により圧縮すると、繰り返し回数と繰り返されるイメージデータの組合せを並べて表現される。即ち、ここでは“AA”というデータが4回繰り返されているため、4に相当する2値データ“03”を付加して、最初の圧縮データは“03AA”となる。また、次の“OF”というデータは1回限りであるから、その圧縮後データは“00OF”となる。それに続く“FO”というデータは“00FO”というデータになる。

【0015】図3には、TIF圧縮方法の例説明図を示す。この図に示した処理対象データは、図2に示した圧縮前の処理対象データと同一の内容である。TIF

圧縮方法においては、繰り返し数の負の値とイメージデータの組によって繰り返されるデータを表現する。また、繰り返されるデータの後に続く繰り返されないデータ列については繰り返さないバイト数-1と繰り返さないイメージデータの列によって表現される。なお、繰り返し数の負の値としたのは、繰り返されるデータの場合に、その繰り返し回数を負の値で表現し、繰り返さないイメージデータに対してはそのイメージデータの数を正で表すようにして、各データの区別ができるようにしたためである。なお、更に具体的な説明は具体例3で行う。

【0016】図4には、Delta-Row圧縮方法の例説明図を示す。この方法は、図2や図3を用いて説明した方法と異なり、直前のラスタライン（前列）のデータとこれから圧縮しようとする処理対象ラスタラインのデータとを比較し、同一の部分と異なる部分を求める。この圧縮方法は、特に同一内容のラスタラインが繰り返し入力するような場合にそのデータを効率よく圧縮できる。

【0017】この処理対象データの内容は図2や図3に示したものと同一である。そして、直前のラスタラインのデータは、図の前列部分に示すように、2バイト目と5バイト目及び6バイト目が処理対象データと異なっている。こうした場合に、この圧縮方法では、異なる部分のデータ列を、コントロールバイトと、置き換えるためのイメージデータの組合せで表す。これを2値の8ビットデータで表現した場合に、上位3ビットは置き換えるバイト数-1で表す。また、下位5ビットは現在位置からのオフセットを表す。こうして得られた8ビットの2値データを16進法のデータで表現すると、この図に示すようになる。即ち、1バイトだけ置き換える場合には、コントロールバイトは“01”となり、置き換えるべきイメージデータ“AA”がこれに続く。また、2バイトのデータが置き換えられるとき、その現在位置からのオフセットを加味してコントロールバイトは“22”となり、置き換えられるべきイメージデータは“OF”と“FO”となる。

【0018】以上の圧縮方法はいずれもよく知られたプリンタ用の圧縮方法で、これらの中から最適の圧縮方法をラスタライン毎に選択していく。この場合に、本発明は次のような方法によってその圧縮方法選択を行う。

【0019】図5には、(a)に本発明による圧縮方法を示し、(b)に比較例による圧縮方法を示した。図において、図1に示した本発明の装置による方法では、まずステップS1～ステップS3で、予め用意されたn種類の圧縮方法による圧縮後のバイト数を所定の演算式を用いて算出する。そして、ステップS4において、どの圧縮方法を用いれば圧縮後のデータ列が最短になるかを調べる。こうして選択された最適の圧縮方法で、実際に処理対象データが圧縮される（ステップS4）。一方、

従来行われた比較例による圧縮方法では、ステップS1～ステップS3において、各圧縮方法により実際に処理対象のデータを圧縮する。そして、圧縮後のデータをメモリに格納する一方、ステップS4において、圧縮後のバイト数を比較する。こうして、最適の圧縮方法が選択され、メモリに格納されたいずれかのデータが出力される。

【0020】このように、本発明において、実際にデータを圧縮することなく圧縮後のデータ長を計算して予測するようにすれば、実際に圧縮を行う場合に比べて簡単な情報を処理対象データから抽出し、しかも計算の過程では圧縮後のデータを一時保存するほど大きなメモリスペースを必要としない。従って、全体として高速に短時間に演算と比較処理ができ、小容量のバッファメモリで処理が可能となる。

【0021】図6は、実際に図1に示した予備検査部2が取得するデータの説明図を示す。(a)は、RLE圧縮及びTIF圧縮の際の圧縮前のデータから圧縮後バイト数を計算するために得るデータを示す。図に示すように、圧縮前のデータの長さがLである場合に、繰り返されないデータと繰り返されるデータとが交互に存在する。このとき、繰り返されないデータの数 m_1, m_2, \dots, m_{x+1} と繰り返されるデータの数 n_1, n_2, \dots, n_x が存在する。

【0022】図の(b)は、Delta-Row圧縮の圧縮前のデータから取り出すべき結果の説明図である。図に示すように、前ラインと同一データは m_1, m_2, \dots, m_{x+1} 、異なるデータは n_1, n_2, \dots, n_x 、データ長はLといった値を求め、演算処理を行うことによってどの圧縮方法が最適かを比較できる。なお、図2～4に示した圧縮方法がなされる場合に、図6に示したような情報を得ることによって、各種の計算方法で実際に圧縮を行うことなくデータ長が演算可能であるが、以下の具体例において、それぞれ簡単に高速に演算が可能な演算方法を紹介する。

【0023】〈具体例2〉図7に、RLE圧縮後バイト数計算式説明図を示す。既に、図6(a)を用いて説明したように、処理対象データ列において、繰り返しのない部分のバイト数が m_1, m_2, \dots, m_{x+1} である場合に、これらの合計をMとする（図7(a)）。また、繰り返しがされる部分のバイト数を n_1, n_2, \dots, n_x とした場合に、これらの合計をNとする（図7(b)）。こうすれば、図7(c)に示すように、ラスタライン全体のバイト数Lは、 $M+N$ によって求めることができる。

【0024】ここで、圧縮後のバイト数を計算する式を図7(d)に示す。まず、 n_1 個のデータが繰り返された場合には、基本的には1バイトの係数によって繰り返し数を表現し、もう1バイトによって繰り返されるデータを表現する。しかしながら、繰り返し数が257回以

上になると、1バイトではその回数を表現することはできない。即ち、257バイト以上繰り返される場合には、繰り返し数と繰り返されるデータの組をつなぎ合わせて表現する。そこで、何組つなぎ合わせるかを計算する式が図に示す $INT(\cdot)$ という部分である。これは、繰り返し部分を表現するための係数の補正分である。ここでは、繰り返し数 $n1$ が256以下の場合には“1”となり、257以上512以下の場合には“2”となる。512を越えた場合にも順にこういった要領で $INT(\cdot)$ の部分が表現される。他の繰り返し部分についても全く同様である。なお、繰り返さない部分については1バイトの“0000”という値と繰り返さないデータとの組合せによって表現されることから、図に示すように $L-N$ を2倍したものとなる。

【0025】この図の(d)に示すように、 $n1 \sim nx$ 及び L と N とを求めることによって圧縮後のバイト数 Z を正確に計算することができる。なお、更に演算処理を容易にするために、図の(e)には(d)の式を簡略化したものを示す。この図によれば、最初の $INT(N/256) \times 2$ という項は(d)に示した繰り返し部分の表現に使う式をくくったもので、繰り返し部分の合計バイト数 N を用いて繰り返し部分を表現するための係数の補正分を表している。次の第2項は $X \times 2$ で、繰り返し部分のブロック数である。256バイト以上繰り返し部分がなければこの項のみで繰り返し部分が表現できる。次の $(L-N) \times 2$ は、(d)に示した繰り返さない部分の式をそのまま使用したものである。

【0026】なお、第1項目の式は繰り返し部分の合計バイト数 N を用いて一括して係数の補正分を表したため若干の誤差が生じる。しかしながら、最小限このバイト数に圧縮されるという結果が得られることから、他の圧縮方法との比較が可能になる。こうすれば、図に示すように繰り返し部分の合計バイト数即ち合計データ長である N と、繰り返し部分のブロック数 X と、ラスタライン全体のバイト数 L によって簡単な式で圧縮後のバイト数を計算することができる。 N 、 X 、 L を求めるための検査は高速かつ容易である。従って、従来のように実際に圧縮処理を行う場合に比べて極めて高速に演算処理ができる。しかも、この N 、 X 、 L の値は他の圧縮方法による圧縮後のバイト数を予測するためにも利用できるため、データの検査は一部省略できるという効果もある。なお、 N 、 X 、 L を求める代わりに M 、 X 、 L を求めるようにしてもよいし、これらの値を別の方法により間接的に求めてもよい。

【0027】図8に、RLE圧縮後のバイト数算出法フローチャートを示す。上記の計算は具体的にはこの図に示すような手順によって行われる。なお、この処理を行うために、データ読み込みポインタ SRC とデータ数カウンタ $COUNT$ と、圧縮データ数カウンタ Z 、繰り返し部分バイト数カウンタ N 、繰り返し部分ブロック数カウ

ンタ X 、繰り返し部分中フラグ F とが用意される。このように、この計算のためには小容量の簡単なバッファのみが使用される。

【0028】なお、データ読み込みポインタ SRC は、処理対象データ列の何番目のデータを読み込んだかを示すポインタである。また、データ数カウンタ $COUNT$ は最初に L をセットオフしておき、1ライン分全てのデータについて処理が終了したかを判断するためのパラメータである。 Z 、 N 、 X は、既に説明したとおりのもので、繰り返し部分中フラグ F は、ポインタがデータの繰り返し部分にくると“1”になり、繰り返し部分外では“0”になるパラメータである。

【0029】図8のステップ $S1$ において、まず繰り返し部分バイト数カウンタ N と繰り返し部分ブロック数カウンタ X とをゼロクリアし、フラグ F も“0”にリセットする。そして、ステップ $S2$ において、まず隣接データ即ち直前のデータと次のデータとの比較を行い、これらが等しいかどうかを判断する。そして、等しければステップ $S3$ に進み、フラグ F が“0”かどうかを判断する。フラグ F が“0”でない場合にはステップ $S4$ に移り、フラグ F を“1”にセットして、ステップ $S5$ で繰り返し部分ブロック数カウンタ X をインクリメントする。即ち、初めて繰り返しブロックが検出されると、そのブロック数 X がカウントされる。更に、ステップ $S6$ において、繰り返しバイト数がインクリメントされる。

【0030】既に繰り返し部分の先頭が検出され、繰り返し数のみをカウントする場合には、ステップ $S3$ からステップ $S6$ にジャンプして繰り返し部分バイト数のみがインクリメントされる。ステップ $S7$ ではポインタを“1”だけインクリメントし、ステップ $S8$ ではカウンタをデクリメントする。ステップ $S9$ で、カウンタが“1”かどうかを判断する。こうして、全ての処理対象データについて処理を終了したかどうかを判断する。

【0031】終了していなければ、再びステップ $S2$ に移る。直前のデータと次のデータとが異なる場合には、ステップ $S2$ からステップ $S13$ に進み、フラグ F が“0”にセットされる。こうして、全てのデータについて、繰り返し部分のブロック数 X と繰り返し部分のバイト数 N とを検出する。そして、これらの処理を終了するとステップ $S10$ に移り、最終データとその直前のデータとが等しかったかどうかを判断する。等しかった場合には、最終データの分について、繰り返し部分バイト数 N をインクリメントする。等しくない場合には、ステップ $S11$ をパスし、ステップ $S12$ に進む。こうしてステップ $S12$ において、上記図7に示した演算処理を実行する。

【0032】〈具体例2の効果〉以上の演算方法により、処理対象データから簡単な情報を得ることによって、正確な圧縮後のデータサイズを予測することができる。

【0033】〈具体例3〉図9は、T I F F圧縮後バイト数計算式の説明図である。この図の(a)、(b)、(c)は、既に図7を用いて説明した繰り返さない部分のバイト数M、繰り返し部分のバイト数N及びラスタライン全体のバイト数Lを示す。ここで、この具体例では、(d)に示すようにして、圧縮後のバイト数を計算する。図の最初の項のINT(・・)は、128バイト以上同一データが繰り返された場合の繰り返し数とデータとの組の数を表す。図7の例では256であったのに、この例では128となったのは次の理由による。

【0034】T I F F圧縮処理においては、8バイトのデータによって表現できる256の数字を前半の128と後半の128とに分ける。そして、前半を0～127まで表現し、後半を-1～-127で表現するように割り当てる。これによって、負の係数で繰り返し数を表し、正の係数で繰り返さないデータのバイト数を表す。その結果、最大128バイトの連続しか表現できないためこの図9(d)に示すような係数長の補正が行われる。その他の原理は図7を用いて説明したものと同一である。なお、繰り返さない部分については、繰り返さない部分のバイト数と繰り返さない部分のデータの数m1とによって表現されるから、第3項目、第4項目のような結果となる。(e)には、圧縮後のバイト数を簡略化した計算式を示した。

【0035】図9の(e)式に示すように、簡略化した式は5項目の和により圧縮後のバイト数Zを求める。最初の項INT(N/128)×2は具体例2の図7に示したケースと全く同様にして、図9(d)の繰り返し部分の係数をまとめたものである。従って、この式は繰り返し部分のバイト数Nのみを用いて求めることができる。これによって、繰り返し部分の表現に用いる係数の係数長の補正分が表される。次の第2項、X×2は繰り返し部分のブロック数を2倍したもので、繰り返し部分が128バイト以下のものばかりであればこの項のみで表現できる。次のINT(L-N/128)は、繰り返さない部分を表現する係数の成分である。次のX+1は繰り返さない部分のブロック数である。繰り返さない部分の全てのブロックが128バイト以下の場合にはこの項だけで係数が表現できる。L-Nは繰り返さない部分の合計バイト数である。この計算も具体例2で求めた繰り返し部分のバイト数N、ラスタライン全体のバイト数L、繰り返し部分のブロック数Xの3種のデータによって簡単に演算できる。その他の係数や算出用データの取扱いは具体例2を用いて説明したものと同様である。

【0036】図10には、T I F F圧縮後のバイト数算出方法フローチャートを示す。この図に示す処理は、具体的には図8に示す処理とほぼ同様となる。即ち、両者を比較してわかるように、その差はステップS12における集計部分のみである。このように異なる圧縮方法を用いた場合の圧縮後のバイト数をほぼ共通のプログラム

を用いて算出することにより、極めて高速に圧縮後のバイト数予測が可能となる。従って、従来のように、それぞれ別々の異なる圧縮方法を用いて具体的に圧縮処理した後バイト数をカウントする方法に比べて著しく圧縮結果の比較演算を高速化することができる。

【0037】〈具体例4〉図11に、Delta-Row圧縮後バイト数計算式説明図を示す。この例においては、(a)に示すように、前列と同一部分のバイト数Mと、(b)に示すように、前列と異なる部分のバイト数Nと、(c)に示すラスタライン全体のバイト数Lとを求める。前列と同一部分の合計バイト数Mは、既に図6を用いて説明したとおり、m1+m2+mx+1となる。また、前列と異なる部分の合計バイト数Nはn1+n2+…+nxとなる。ここで、圧縮後のバイト数Zは、

(d)に示すとおり演算される。(d)の式の第1項は異なる部分を示し、図4に示したように置換バイト数を3ビットで表すため、8個以上の場合には、このデータが2組以上繰り返される。そこで、INT(・・)の部分がこの図に示すように補正用に演算される。また、同一部分については、第3番目、第4番目の項及びn1～nxまでのバイト数のデータが付け加えられる。(e)に示したものは、これらを簡略化した演算式である。これも具体例2や具体例3と同様に高速演算のために簡略化を行い、圧縮後のバイト数Zの最大値を求めるようにしている。

【0038】図11の(e)において、第1項目は異なる部分の長さを表現するための係数長の補正分である。Xは前列と異なる部分のブロック数で、第3項目はオフセット位置を示すための係数長補正分である。これに前列と異なる部分のブロック数Xとそのバイト数Nを加算する。このように(d)式を簡略化し、最小限このバイト数に圧縮されるという結果を得て他の圧縮方式と比較する。なお、この例では、M、Nは具体例2、具体例3と別の値であって、別の方法によって検査し求めることになる。しかしながら、やはりM、N、Lといった簡単な数値を用いて演算処理を行うため高速で圧縮後のバイト数算出が可能となる。

【0039】図12には、Delta-Row圧縮後のバイト数算出法フローチャートを示す。ここでは、新たに前列データポインタSEEDをセットし、前列と圧縮対象データ列とを比較しながら処理を行う。ステップS2、S3、S4、S5、S6及びステップS12の処理は、具体例2、具体例3と同様である。また、ステップS7では、前列と処理対象データ列とのポインタを同期させるために、前列データポインタSEEDをインクリメントしている。ステップS8、S9、S10は、これまでの具体例の処理と同様である。こうして、ステップS11において、比較処理に必要な圧縮後のバイト数Zを得る。

【0040】〈具体例4の効果〉この具体例において

も、実際に処理対象データ列を圧縮処理することなく圧縮処理後のデータ長を高速演算処理することが可能になる。

【図面の簡単な説明】

【図1】本発明の装置の具体例を示すブロック図である。

【図2】RLE圧縮方法の例説明図である。

【図3】T I F F圧縮方法の例説明図である。

【図4】Delta-Row圧縮方法の例説明図である。

【図5】本発明による圧縮方法と比較例による圧縮方法のフローチャートである。

【図6】各圧縮方法の圧縮前のデータから得る情報の説明図である。

【図7】RLE圧縮後バイト数計算式説明図である。

【図8】RLE圧縮後のバイト数算出法フローチャートである。

【図9】T I F F圧縮後バイト数計算式説明図である。

【図10】T I F F圧縮後のバイト数算出法フローチャートである。

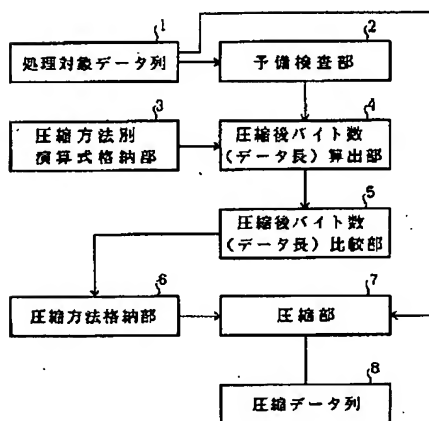
【図11】Delta-Row圧縮後バイト数計算式説明図である。

【図12】Delta-Row圧縮後のバイト数算出法フローチャートである。

【符号の説明】

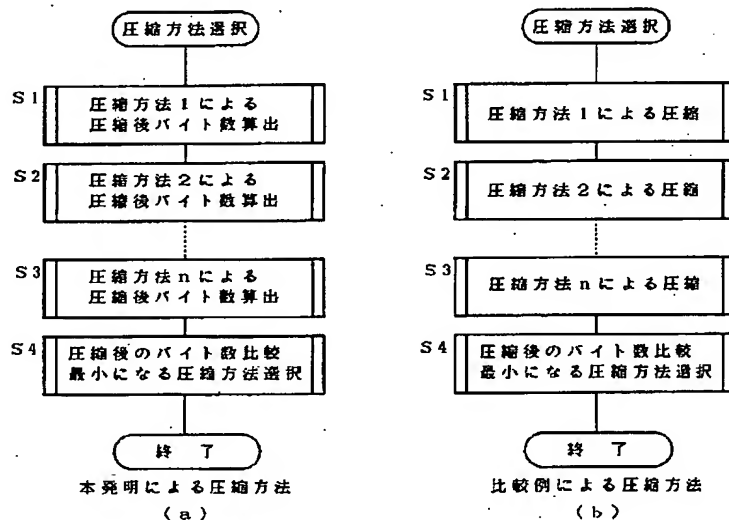
- 1 処理対象データ列
- 2 予備検査部
- 3 圧縮方法別演算式格納部
- 4 圧縮後バイト数算出部
- 5 圧縮後バイト数比較部
- 6 圧縮方法格納部
- 7 圧縮部
- 8 圧縮データ列

【図1】

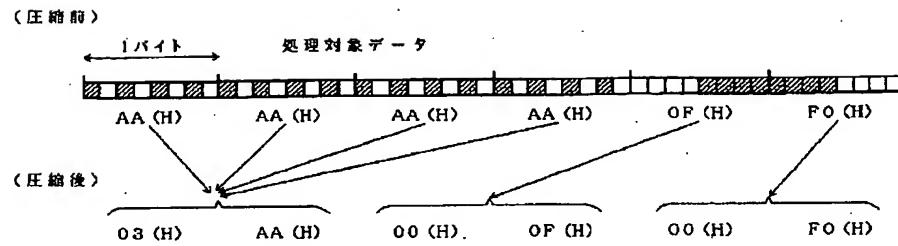


本発明の装置の具体例ブロック図

【図5】

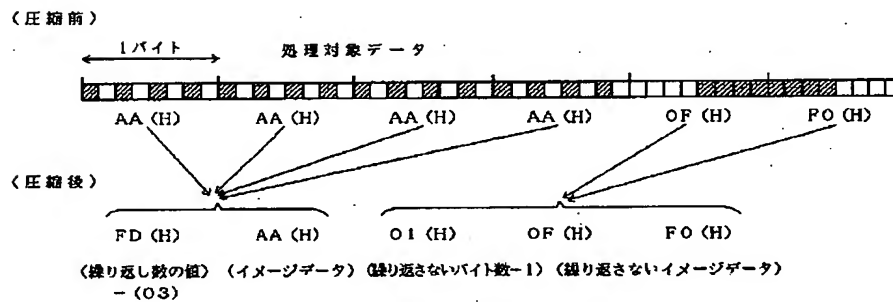


【図2】



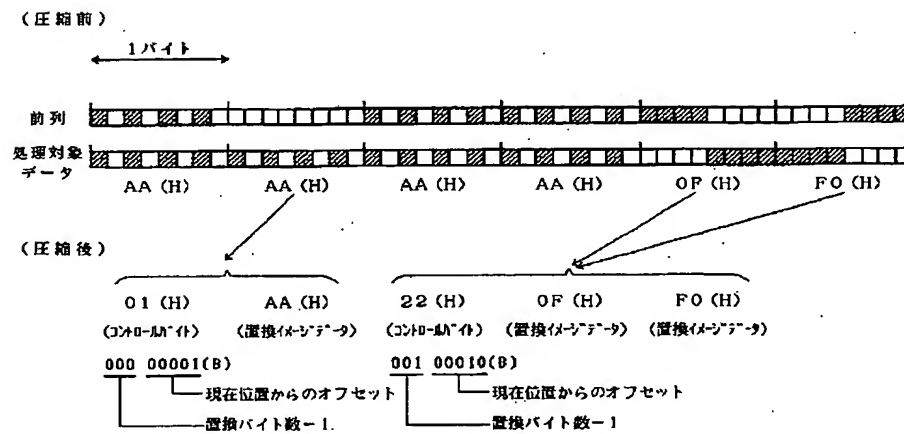
R L E 圧縮方法の例説明図

【図3】



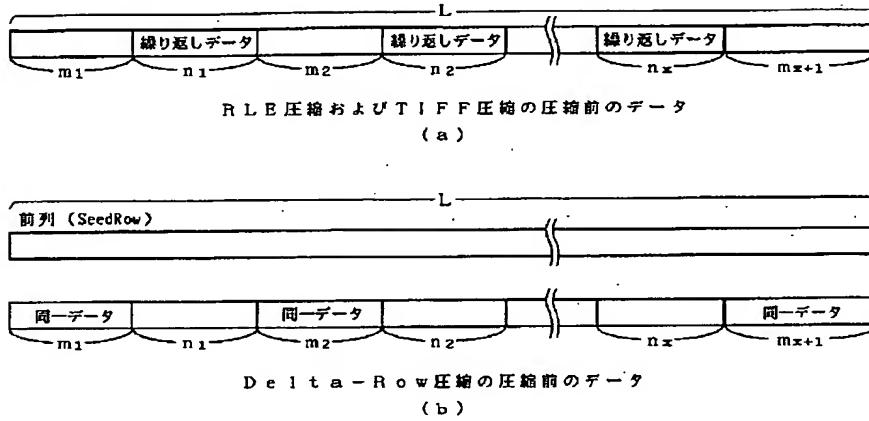
T I P F 圧縮方法の例説明図

【図4】



Delta-Row 圧縮方法の例説明図

【図6】



【図7】

- (a) 繰り返さない部分のバイト数 $M = m_1 + m_2 + \dots + m_{x+1}$
 (b) 繰り返す部分のバイト数 $N = n_1 + n_2 + \dots + n_x$
 (c) ラスタライン全体のバイト数 $L = M + N$
 (d) 圧縮後のバイト数

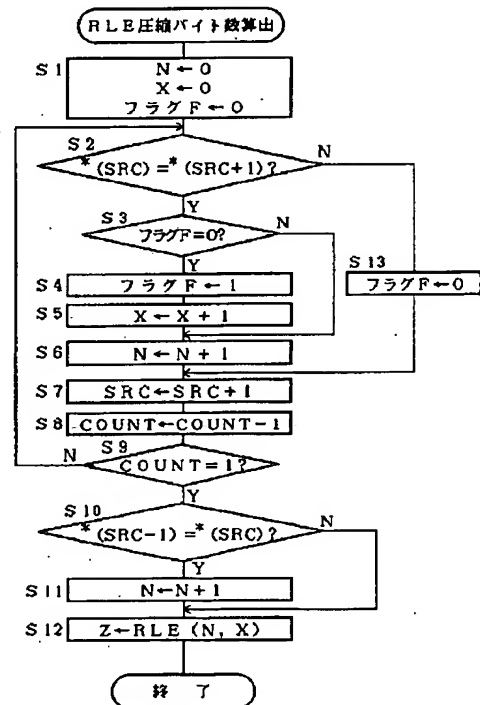
$$\begin{aligned}
 Z &= \text{INT} \left(\frac{n_1 + 256}{256} \right) * 2 && n_1 \text{個繰り返し分} \\
 &+ \text{INT} \left(\frac{n_2 + 256}{256} \right) * 2 && n_2 \text{個繰り返し分} \\
 &\dots\dots\dots \\
 &+ \text{INT} \left(\frac{n_x + 256}{256} \right) * 2 && n_x \text{個繰り返し分} \\
 &+ (L - N) * 2 && \text{繰り返さない分}
 \end{aligned}$$

- (e) 圧縮後のバイト数 (簡略式)

$$\begin{aligned}
 Z &\leq \text{INT} \left(\frac{N}{256} \right) * 2 && 256 \text{バイト以上の繰り返し分} \\
 &+ X * 2 && \text{繰り返し分} \\
 &+ (L - N) * 2 && \text{繰り返さない分}
 \end{aligned}$$

R L E 圧縮後バイト数計算式説明図

【図8】



R L E 圧縮後のバイト数算出法

【図9】

- (a) 繰り返さない部分のバイト数 $M = m_1 + m_2 + \dots + m_{x+1}$
 (b) 繰り返す部分のバイト数 $N = n_1 + n_2 + \dots + n_x$
 (c) ラスタライン全体のバイト数 $L = M + N$
 (d) 圧縮後のバイト数

$$Z = \text{INT} \left(\frac{n_1 + 127}{128} \right) * 2 \quad \text{繰り返し分}$$

$$\dots\dots\dots$$

$$+ \text{INT} \left(\frac{n_x + 127}{128} \right) * 2 \quad \text{繰り返し分}$$

$$\dots\dots\dots$$

$$+ \text{INT} \left(\frac{m_1 + 127}{128} \right) + m_1 \quad \text{繰り返さない分}$$

$$\dots\dots\dots$$

$$+ \text{INT} \left(\frac{m_{x+1} + 127}{128} \right) + m_{x+1} \quad \text{繰り返さない分}$$

- (e) 圧縮後のバイト数 (簡略式)

$$Z \leq \text{INT} \left(\frac{N}{128} \right) * 2 + x * 2 \quad \text{繰り返し分}$$

$$+ \text{INT} \left(\frac{L - N}{128} \right) + (x + 1) \quad \text{繰り返さない分}$$

$$+ L - N \quad \text{繰り返さない分}$$

T I F F 圧縮後バイト数計算式説明図

【図11】

- (a) 前列と同一部分のバイト数 $M = m_1 + m_2 + \dots + m_{x+1}$
 (b) 前列と異なる部分のバイト数 $N = n_1 + n_2 + \dots + n_x$
 (c) ラスタライン全体のバイト数 $L = M + N$
 (d) 圧縮後のバイト数

$$Z = \text{INT} \left(\frac{n_1 + 7}{8} \right) \quad \text{異なる部分}$$

$$\dots\dots\dots$$

$$+ \text{INT} \left(\frac{n_x + 7}{8} \right) \quad \text{異なる部分}$$

$$\dots\dots\dots$$

$$+ \text{INT} \left(\frac{(m_1 - 30) + 254}{255} \right) \quad \text{同一の部分}$$

$$\dots\dots\dots$$

$$+ \text{INT} \left(\frac{(m_x - 30) + 254}{255} \right) \quad \text{同一の部分}$$

$$+ n_1 + n_2 + \dots + n_x$$

- (e) 圧縮後のバイト数 (簡略式)

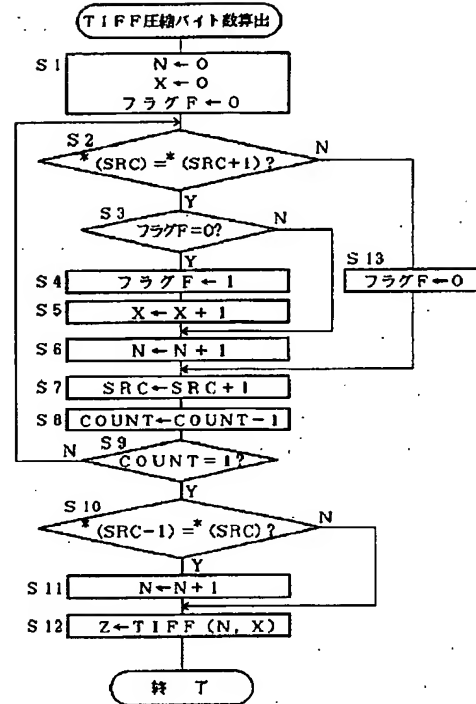
$$Z \leq \text{INT} \left(\frac{N}{8} \right) + x$$

$$+ \text{INT} \left(\frac{M - 30 * x}{255} \right) + x$$

$$+ N$$

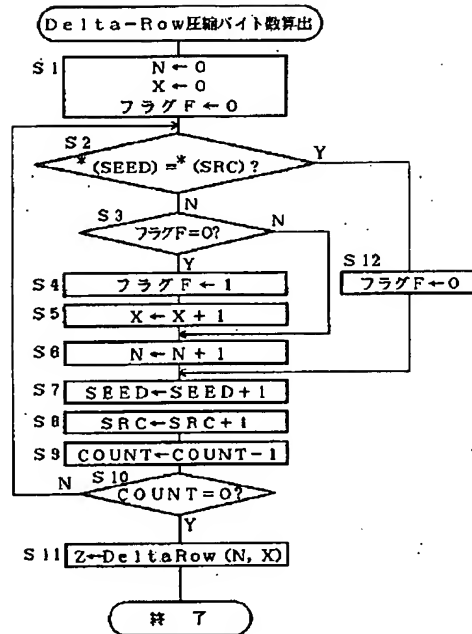
Delta-Row圧縮後バイト数計算式説明図

【図10】



T I F F 圧縮後のバイト数算出法

【図12】



Delta-Row圧縮後のバイト数算出法